17

Multi-Objective Scheduling on a Single Machine with Evolutionary Algorithm

A. S. Xanthopoulos, D. E. Koulouriotis and V. D. Tourassis
Democritus University of Thrace
Greece

1. Introduction

In this chapter, the single-machine scheduling problem (SMSP) is addressed by adopting a multi-objective perspective. A finite set of independent jobs has to be scheduled on a single server that is continuously available and has the ability to process only one job at a time. Once a job has started its processing it cannot be interrupted and no idle time between successive jobs is allowed. Each job is characterized by its processing time and due date, both integer numbers.

Job scheduling considering only one criterion can be thought of as an over-simplification of the problem, since there are several objectives related to this problem, namely sum of earliness/tardiness, max earliness/tardiness and so forth. There is a growing trend in the relevant literature to study scheduling problems under multiple objectives. Some indicative examples can be found in the works of (Behnamian et al., 2009); (Yagmahan et al., 2010); (Loukil et al., 2005); (Moslehi & Mahnam, 2010); (Choobineh et al, 2006) and (Gupta & Sivakumar, 2005).

In this chapter we consider two objectives; sum of earliness and sum of tardiness. In general, these objectives are conflicting meaning that a solution that improves one objective function will deteriorate the other. In the absence of optimization criteria preferences each one objective needs to be dealt with explicitly, giving rise to the concept of Pareto optimality. The solution to the bi-objective single-machine scheduling problem is a set of points in objective space known as Pareto front and all points along the Pareto front share the following property: the value of a certain objective function can be improved only by degrading the value of at least one of the remaining objective functions.

The derivation of the Pareto set using exhaustive search is limited to relatively small instances of the problem due to the combinatorial explosion that takes place as the dimensionality increases. Consequently, the application of meta-heuristics appears to be well-suited for this type of problems. The reader is referred to (Bagchi, 1999) for further information on multi-objective scheduling using evolutionary techniques. In this chapter, a multi-objective algorithm evolutionary algorithm (MOEA) for approximating the Pareto front in single-machine scheduling problems is described and analyzed. The MOEA is tested in a series of randomly generated SMSP instances of various configurations. In order to select appropriate parameters for the MOEA we apply techniques from the field of design of experiments (DoE), specifically general fractional factorial designs. The solutions found by the MOEA are

compared to the true Pareto front and to each other based on four metrics: generational distance, non-uniformity of solution distribution, hypervolume and maximum spread.
The structure of this chapter is presented hereafter. In Section 2 the multi-objective version of the SMSP is described. Section 3 is devoted to the presentation of the salient features of SPEA2, the MOEA which was applied to solve the underlying optimization problem. Four metrics for comparing Pareto sets are presented in Section 4. The numerical results from the series of experiments are presented and commented upon in Sections 5 to 5.5. Finally, Section 6 contains the concluding remarks and some directions for future research.

2. Problem description

The formal description of the problem under consideration is given in this section. Let $J = \{j_1, j_2, ..., j_n\}$ be a finite set of jobs. The assumptions pertaining to the set J are the following.

- each job constitutes an indivisible whole, i.e. it cannot be broken down to elementary operations.
- the release time of the *i*-th job is denoted as r_i and for all *i*, $r_i = 0$, meaning that all jobs are available for processing from $t = 0$.
- each job is associated with a non-negative integer processing time $p_i \in Z^+$.
- each job has a due date $d_i \in Z$. Due dates can assume negative values and the meaning of a negative due date is that the related job is already delayed.

All jobs in J have to be processed on a single machine to which, the following assumptions apply.

- the machine can only process one job at a time.
- once the machine has started processing a job it cannot be interrupted.
- the machine does not undergo failures nor does it suspend its operation for maintenance or other reasons.
- once a job has completed its processing it exits the system immediately, and as a consequence, the machine is never blocked.

The underlying decision problem is to determine the sequence according to which all jobs will be processed on the machine under two additional assumptions:

- the machine setup time for shifting from one job to another is always zero.
- no machine idle time between successive job is allowed.

The sequence of jobs is called a *schedule* and a possible schedule is a permutation of the elements that belong to the set $\{1,2,...,n\}$. Clearly, an admissible schedule is a *n*-dimensional vector **s** where s_i is the position of the *i*-th job in the schedule and the number of all plausible schedules is $n!$. For a given schedule, the completion time of the *i*-th job is $c_i = w_i + p_i$ where w_i is the waiting time related to that job. The waiting time of job j_i is the sum of the processing times of all jobs that were completed up to the point that j_i starts its processing:

$$w_i = \sum_{J_i^p} p_k \ , \ J_i^p = \{j_k \in J : s_k < s_i\} \tag{1}$$

The *earliness* of the *i*-th job is $e_i = \max\{0, d_i - c_i\}$ and the *tardiness* of job j_i is $t_i = \max\{0, c_i - d_i\}$.

The single-objective approach to compare two alternative schedules $\mathbf{s}$ and $\mathbf{s}'$ is to define an objective metric f that would assign a numerical value $f(\mathbf{s})$ to every $\mathbf{s} \in S$, where S is the set of possible schedules. However, as stated in the introduction, more that one objective functions can be associated with the problem under consideration. In this chapter two objective functions are considered simultaneously, forming the objective vector described in (2).

$$\mathbf{f} = [f_1, f_2] = [\sum_{i=1}^{n} e_i, \sum_{i=1}^{n} t_i] \tag{2}$$

The first element of the objective vector is the sum of earliness values of all jobs whereas the second objective element is the sum of all tardiness values. Both of these two quantities are to be minimized. The adoption of the two objective functions for quantifying the system's performance is compatible with the philosophy of Just In Time manufacturing, according to which an end-item should be ideally complete its processing exactly at the time when it is needed. In the total absence of objective function preferences each objective must be dealt with explicitly. The concept of Pareto dominance can be used to compare two candidate solutions in the multi-objective setting where each objective component is treated separately. In the minimization problem treated in this chapter, an objective vector $\mathbf{f}_a$ dominates another vector $\mathbf{f}_b$, *iff*

$$f_{a,i} \leq f_{b,i}, \forall i \in \{1,2\} \tag{3}$$

and

$$\exists j \in \{1,2\} \text{ such that } f_{a,j} < f_{b,j} \tag{4}$$

Pareto dominance in this case is denoted by $\mathbf{f}_a \prec \mathbf{f}_b$. Using the notion of Pareto dominance, the objective functions that constitute the objective vector are characterized as *partially conflicting*, meaning that there is at least one decision vector $\mathbf{s}$ dominated by some other vector that belongs to S.

The solution to the multi-objective optimization problem stated in this section is the global Pareto optimal set P, that is, the set of objective vectors which are not dominated by any other feasible objective vector.

3. SPEA2

Some examples of popular multi-objective evolutionary algorithms (MOEAs) are PESA (Corne et al. 2000), PESA-II (Corne et al., 2001), SPEA (Zitzler & Thiele, 1999), SPEA2 (Zitzler et al., 2001), NSGA-II (Deb et al., 2002), MOEA (Tan et al., 1999), ESPEA (Everson et al. 2002), DMOEA (Lu & Yen, 2002) and μGA2 (Pulido and Coello Coello, 2003). The performance of a MOEA is typically assessed on the basis of its ability to approximate effectively the true global Pareto front and to produce a uniformly distributed set of solutions in addition to its consistency and robustness. The SPEA2 algorithm has been shown in (Tan et al, 2005) and (Zitzler et al., 2002) to perform very well in comparison to other MOEAs in a variety of different test problems and under several MOEA-specific performance metrics, and was therefore adopted as the search algorithm for the purposes of this investigation. The key features of SPEA2 are outlined in the remaining of this section. For a detailed description of the SPEA2 algorithm the reader is referred to (Zitzler et al., 2002).

The SPEA2 algorithm evolves a population of candidate solutions while maintaining an external population called *archive* where non-dominated individuals found during the evolutionary process are stored. The population is initialized with randomly generated individuals and its size remains constant throughout the execution of the algorithm. Initially the archive is empty but after the first generation of candidate solutions is evaluated the archive is resized to contain a pre-specified number of individuals and its size is kept fixed hereinafter. In each iteration, all individuals in the population and the archive are assigned a fitness value. The SPEA2 algorithm employs a sophisticated fitness assignment scheme, where the fitness of an individual is given by the sum of the *strengths* of its dominators plus its *density*. The strength of an individual represents the number of solutions that it dominates, while the density of an individual is the inverse of the distance from its *k*-th nearest neighbour (another individual) in objective space. After all elements in the population and the archive have been assigned a fitness value, the non-dominated individuals are copied to the *temporary archive*. If the size of the temporary archive exceeds the pre-specified threshold then it is truncated, whereas if the size of the archive is smaller than the threshold then it is expanded by adding to it the best (in terms of fitness value) dominated individuals from the population and the archive. For the truncation of the temporary archive a sequential procedure is applied, where in each pass of the procedure the element with minimum distance (in objective space) to another individual is deleted. Ties are broken by considering the second smallest distance, the third etc. After the temporary archive reaches the specified size, all of its contents are stored in the archive and the temporary archive is deleted. Subsequently, the selection operator is applied on the archive and the selected individuals are subjected to standard genetic operations in order to produce the next population. The algorithm terminates when the stopping criterion is satisfied and returns the archived non-dominated set of solutions. The pseudo-code for the SPEA2 algorithm is presented below, where $|\cdot|$ denotes cardinality of set and S symbolizes the size of the archive:

1. generate initial population *Pop* and empty archive *Arc*
2. assign fitness values to all individuals in *Pop* and *Arc*
3. copy all non-dominated solutions in *Pop* and *Arc* in temporary archive *TempArc*
   a. IF $|TempArc| > S$
      apply truncation procedure on *TempArc*
   b. IF $|TempArc| < S$
      add dominated individuals from *Pop* and *Arc* to *TempArc* based on fitness
4. set $Arc \leftarrow TempArc$, delete *TempArc*
   a. IF $STOPPING\_CRITERION = TRUE$
      return *Arc*. Terminate
   b. ELSE
      go to Step 5
5. apply selection operator on *Arc*. Apply genetic operators on the selected individuals and store the offspring in *Pop*. Go to Step 2

In the remaining of this section we discuss the implementation of the key features of the SPEA2 algorithm that was applied to the problem addressed in this chapter, namely the encoding of candidate solutions and the selection strategy/genetic operators in Step 5 of the algorithm.

All individuals in the population are encoded as vectors of *random keys*. A random key is simply a real number which assumes values in the range $[0,1]$. In order to translate a vector of random keys to a schedule the elements of the chromosome are sorted in descending order. An example of this decoding scheme involving 4 jobs is depicted below.

jobs:	j_1	j_2	j_3	j_4
chromosome:	0.98	0.12	0.56	0.78
sorted genes:	0.98	0.78	0.56	0.12
schedule:	1	4	3	2

The major advantage of random key encoding is that every possible chromosome decodes to a feasible schedule and therefore, there is no need for customized initialization routines and genetic operators. *Tournament selection* is responsible for selecting individuals from the archive for recombination. According to this selection scheme *TourSize* individuals are chosen with the same probability from the archive and the fittest individual from that subset is chosen to constitute a parent which will be recombined with another individual to produce offspring. Parameter *TourSize* is commonly known as the *tournament size* and tournament selection mechanisms with $TourSize = 2$ are referred to as binary tournaments. The procedure is iterated until the required number of individuals has been selected. The selected individuals are recombined according to the *intermediate recombination* technique. The *i*-the element of the offspring is computed according to the following equation:

$$c_i^o = u_i c_i^{p1} + (1 - u_i) c_i^{p2} \tag{5}$$

where c_i^{p1} and c_i^{p2} are the *i*-th elements of the two parents and u_i is a random variable uniformly distributed in $[-\delta, 1+\delta]$. Parameter δ determines the hypercube to which the produced offspring is possible to fall in. The fraction of the new population which consists of individuals generated via recombination is determined by parameter *RecFrc*, whereas the remaining individuals are produced using *migration*, i.e. they are generated at randomly.

4. Performance metrics for non-dominated sets

A number of metrics for comparing non-dominated sets have been proposed in the relevant literature. These metrics largely fall into two categories: i) performance metrics that require the true Pareto front to be known and, ii) performance measures that do not involve the true Pareto front in the computation and can be used to compare two or more non-dominated sets directly. In this investigation we consider the following four measures: a) the *generational distance GD* (Tan et al., 2005), b) the metric of *non-uniform distribution of solutions U*, c) the *maximum spread MS* of the obtained solutions and, d) the *hyper-volume HV* (Tan et al., 2005). The rest of this section briefly describes these performance metrics.

The metric of generational distance reflects the "distance" between a Pareto optimal set *A* and the true Pareto front *P*. It is defined as:

$$GD = \left(\frac{1}{|A|} \sum_{i=1}^{|A|} d_i^2 \right)^{1/2} \tag{6}$$

where $|A|$ is the cardinality of A and d_i is the Euclidean distance between the i-th element of A and the nearest element that belongs to P.
The metric of non-uniform distribution of solutions $U \in \Re^+$ measures how uniformly the individual solutions are distributed in a Pareto optimal set A. It is defined as:

$$U = \sqrt{\frac{\sum_{i=1}^{|A|-1}\left(d_{i,i+1}/\overline{d}-1\right)^2}{|A|-1}} \tag{7}$$

where $|A|$ is the cardinality of A, $d_{i,i+1}$ symbolizes the Euclidean distance between two successive members in A and $\overline{d}$ is the average distance. Large deviations of the distances $d_{i,i+1}$ from the average distance result in high values of U, therefore a Pareto front with a lower value of this metric is preferable to another front with higher U value.
The performance measure called max spread is an indicator of the range in objective space covered by a Pareto optimal set A. Its formal definition is given in Equation (8):

$$MS = \sqrt{\frac{1}{N}\sum_{j=1}^{N}\left(\max_{i=1}^{|A|} f_j^i - \min_{i=1}^{|A|} f_j^i\right)^2} \tag{8}$$

where $|A|$ is the cardinality of A, N is the number of objective functions and f_j^i is the value of the j-th objective yielded by the i-th element in A. Higher values of MS signify better performance
Hyper-volume HV measures the size of the objective space that is dominated by the elements of a non-dominated set A. It is defined as $HV = volume\left(\bigcup_{i=1}^{|A|} v_i\right)$, where v_i is the hypercube with diagonal corners the objective vector $\mathbf{f}^i = [f_1^i, f_2^i, ..., f_N^i]$ of the i-th element in A and the anti-optimal objective vector $\mathbf{f}^{\max} = [f_1^{\max}, f_2^{\max}, ..., f_N^{\max}]$, where $f_j^{\max} = \max_{i=1}^{|A|} f_j^i$. Again, $|A|$ denotes the cardinality of A, N is the number of objective functions and f_j^i is the value of the j-th objective for the i-th element in A.

5. Results

Section 5 and its subsections are devoted to the presentation of the results that were obtained from the application of the SPEA2 algorithm to twelve instances of the single-machine scheduling problem described in Section 2. The randomly generated instances of the problem which drive the experimental investigation are given in the following subsection.

5.1 SMSP instances

All instances were selected at random from an extensive set of SMSP instances which was generated by (Valente & Goncalves, 2009) and can be found online at http://www.fep.up.pt/docentes/jvalente/benchmarks.html. An SMSP instance is simply a set of processing time-due date pairs and it is identified by using the naming convention adopted by (Valente & Goncalves, 2009) specifically *PV-N-T-R*, where *PV* symbolizes the processing time variability, *N* is the number of jobs, *T* is the tardiness factor and *R* is the due date range. All twelve instances considered in this chapter are of type *H-10-x-y*, meaning that they consist of ten jobs with high (*H*) processing time variability, i.e. the processing

times are uniformly distributed in the interval $[1-100]$. Each job has a due date which is drawn from the uniform distribution defined on the interval $[P(1-T-R/2), P(1-T+R/2)]$, where P is the sum of the processing times of all jobs for a particular instance and parameters P and R assume non-negative real values. In this study we consider three levels for the tardiness factor T, specifically 0.4, 0.6, and 0.8.

H-10-0.4-0.2		H-10-0.4-0.4		H-10-0.4-0.6		H-10-0.4-0.8	
p_i	d_i	p_i	d_i	p_i	d_i	p_i	d_i
73	439	9	236	37	144	64	473
51	411	35	290	65	254	82	290
69	411	59	296	34	206	88	159
82	356	47	180	9	239	9	447
51	353	46	294	18	118	23	378
59	363	38	295	39	226	47	144
41	475	44	291	7	206	30	525
85	390	83	225	47	184	97	388
86	377	61	218	21	162	30	461
89	461	12	222	44	124	67	287

Table 1. Instances *H-10-0.4-y*

Additionally, for each tardiness factor level four due date range levels (0.2, 0.4, 0.6, 0.8) are examined. The parameters of the twelve SMSP instances are shown in Tables 1 to 3.

H-10-0.6-0.2		H-10-0.6-0.4		H-10-0.6-0.6		H-10-0.6-0.8	
p_i	d_i	p_i	d_i	p_i	d_i	p_i	d_i
36	135	62	248	53	310	85	117
54	168	45	228	96	75	32	114
84	115	25	105	71	149	13	197
24	137	21	128	81	332	44	135
34	168	78	127	26	61	60	348
33	146	42	161	13	245	100	282
20	129	14	127	2	314	15	198
31	133	38	91	33	323	3	65
5	162	38	150	97	70	84	240
50	126	53	200	36	190	38	183

Table 2. Instances *H-10-0.6-y*

H-10-0.8-0.2		H-10-0.8-0.4		H-10-0.8-0.6		H-10-0.8-0.8	
p_i	d_i	p_i	d_i	p_i	d_i	p_i	d_i
53	108	4	107	85	212	50	-44
89	90	99	160	98	94	70	-52
59	147	41	115	13	7	45	68
31	70	89	116	46	8	79	168
56	171	45	112	99	9	73	196
91	163	41	216	28	73	68	60
73	191	99	51	36	220	28	44
81	132	49	157	6	211	39	192
54	193	73	94	100	250	12	321
57	182	39	83	19	84	94	118

Table 3. Instances *H-10-0.8-y*

5.2 Configuration of MOEAs

The parameters of the multi-objective genetic algorithm (SPEA2) which was used to approximate the true Pareto front for each instance are

- the population size
- the chromosome length
- the maximum number of iterations
- the size of the archive
- the tournament size (selection operator)
- the recombination fraction
- parameter δ (recombination operator)

Since each instance consists of ten jobs and individuals are encoded as sequences of random keys, the chromosome length is equal to 10 in all executions of the algorithm. The exhaustive search algorithm used to find the true Pareto fronts conducts $10! = 3,628,800$ feasible schedule evaluations for each problem instance. We selected the total number of evaluations performed by the MOEA in each instance to be fixed to the level of 35,000, which is approximately 1% of the number of the exhaustive search evaluations. As a consequence, the adjustment of the population size implicitly determines the maximum number of iterations too. The size of the archive was set to be equal to the size of the true Pareto front for each instance. Moreover, the recombination fraction determines the fraction of individuals created by recombination in a new generation, where the remaining individuals are generated by the migration operator.

As a consequence, the parameters which participate to the fractional factorial experiments conducted for each problem instance are

i. the population size (*PopSize*)
ii. the tournament size (*TourSize*)
iii. the recombination fraction (*RecFrc*)
iv. the parameter delta (δ)

The levels for factor i were set to be 50 and 100, whereas the levels for factor ii were 2 (binary tournament) and 10, or in the case were the arc consisted of less than ten elements, the high level of parameter ii was equal to the arc size. 0.6 was the low level for factor iii (recombination fraction) and 0.8 the high level. Finally, the two levels of parameter δ were 0 and 0.25. After the levels of the factors had been defined a 2^{4-1} fractional factorial design of resolution IV was generated by employing the Franklin-Bailey algorithm (Box et al., 2005). This type of design separates main effects and requires 8 treatments (parameter sets) to be evaluated, i.e. it is considerable more economical than the corresponding full 2^4 factorial experiment which requires 16 runs. The resulting experimental designs are presented in Table 4. For every design the evolutionary algorithm is executed 3 times and the non-dominated set which has the highest number of elements identical to that of the corresponding true Pareto front is selected.

	PopSize	*TourSize*	*RecFrc*	δ
design 1	50	2	0.6	0
design 2	50	2	0.8	0.25
design 3	50	10(archive size)	0.6	0.25
design 4	50	10(archive size)	0.8	0
design 5	100	2	0.6	0.25
design 6	100	2	0.8	0
design 7	100	10(archive size)	0.6	0
design 8	100	10(archive size)	0.8	0.25

Table 4. Experimental designs

5.3 Comparative evaluation – instances H-10-0.4-y

The cardinalities of the true Pareto fronts for instances H-10-0.4-0.2, H-10-0.4-0.4, H-10-0.4-0.6 and H-10-0.4-0.8 are 2, 10, 6 and 3, respectively. For all of these four instances the best

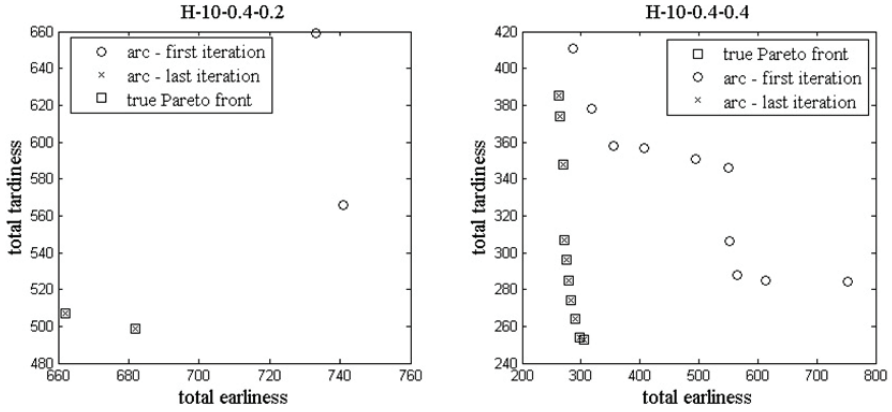


Fig. 1. True Pareto fronts, archives in first/last iteration – instances *H-10-0.4-0.2* and *H-10-0.4-0.4*

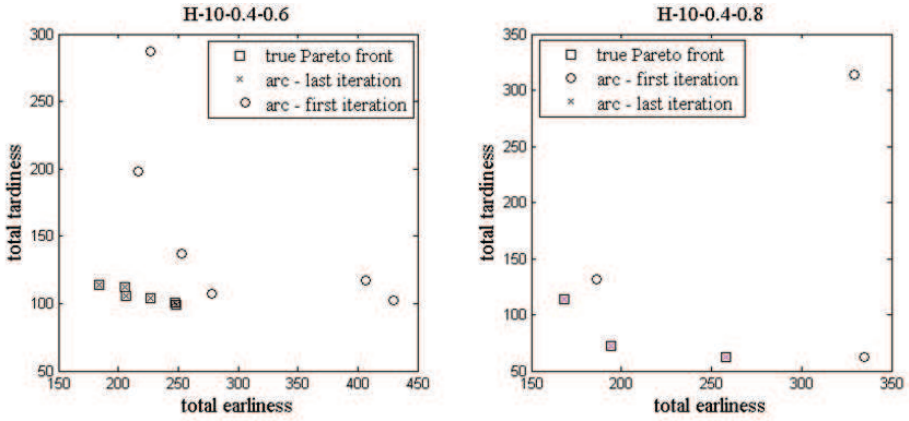


Fig. 2. True Pareto fronts, archives in first/last iteration – instances *H-10-0.4-0.6* and *H-10-0.4-0.8*

design for the MOEA was found to be the third which corresponds to $PopSize = 50$, $RecFrc = 0.6$, $\delta = 0.25$ and *TourSize* set to the high level. The non-dominated sets returned by the MOEAs initialized with that parameter set were identical to the true Pareto fronts for all problem instances examined in this subsection. The non-dominated sets related to executions of the MOEA with different parameter sets were not globally Pareto optimal and therefore further comparisons between the various sets using the metrics presented in Section 4 is redundant. Figures 1 and 2 illustrate the true Pareto fronts of the four instances of this subsection, and the individuals (in objective space) that populate the archive in the first and last iteration of the best evolutionary algorithm execution.

5.4 Comparative evaluation – instances H-10-0.8-y

The true Pareto fronts for instances H-10-0.8-0.2, H-10-0.8-0.4, H-10-0.8-0.6 and H-10-0.8-0.8 consist of 5, 8, 5 and 7 elements, respectively. Similarly to the previous four instances the

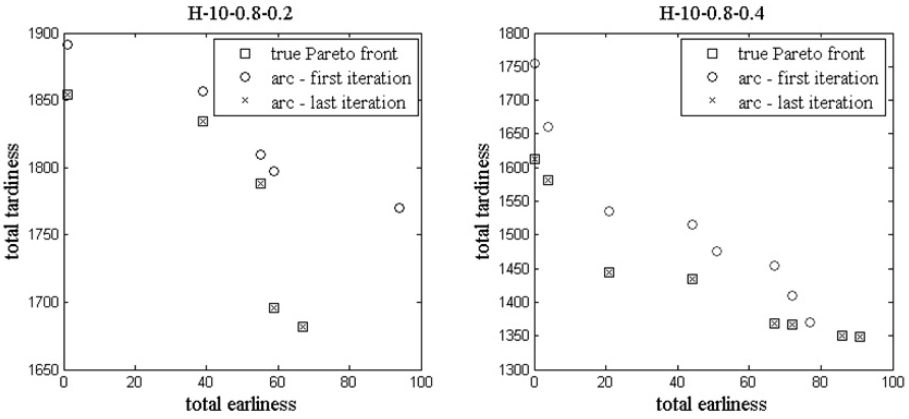


Fig. 3. True Pareto fronts, archives in first/last iteration – instances *H-10-0.8-0.2* and *H-10-0.8-0.4*

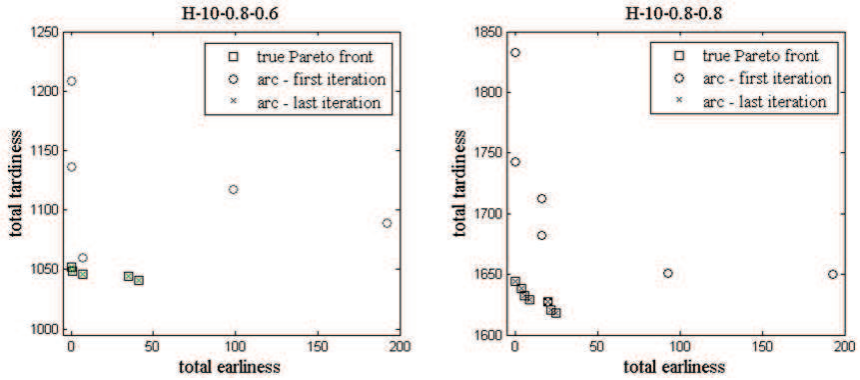


Fig. 4. True Pareto fronts, archives in first/last iteration – instances *H-10-0.8-0.6* and *H-10-0.8-0.8*

best parameter set for the MOEA was $PopSize = 50$, $RecFrc = 0.6$, $\delta = 0.25$ and *TourSize* set to the high level. Executions of the MOEA with those parameters managed to deliver the true Pareto fronts in all four cases whereas executions with alternative parameter sets failed to do so. Figures 3 and 4 present the actual Pareto fronts of the four instances of this subsection, and the individuals that constitute the archive in the first and last generation of the best evolutionary algorithm execution.

5.5 Comparative evaluation – instances H-10-0.6-y

The sizes of the non-dominated sets related to the last four instances of this experimental investigation are generally higher than those of the eight instances discussed in the previous two subsections. More specifically, the actual Pareto fronts for instances H-10-0.6-0.2, H-10-0.6-0.4, H-10-0.6-0.6 and H-10-0.6-0.8 have 13, 10, 14 and 9 elements, respectively. In this subset of problem instances, the true Pareto fronts corresponding to instances H-10-0.6-0.4 and H-10-0.6-0.8 were computed exactly by the MOEA. The parameter values related to the third design of the fractional factorial experiment were found to be the best for this series of problem instances too. In the case of instance H-10-0.6-0.2 the MOEA returned a set of

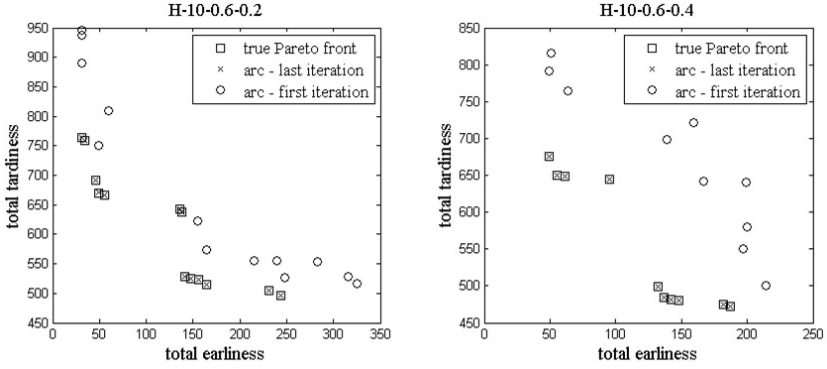


Fig. 5. True Pareto fronts, archives in first/last iteration – instances *H-10-0.6-0.2* and *H-10-0.6-0.4*

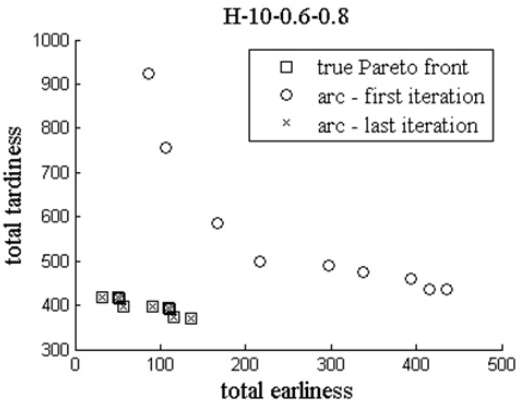


Fig. 6. True Pareto front, archive in first/last iteration – instance *H-10-0.6-0.8*

non-dominated solutions that is differentiated from the actual Pareto front only in one element, where the globally Pareto optimal element is $[49,670]$ and the corresponding element of the MOEA front is $[49,672]$. It is reasonable to argue that the approximation of the actual Pareto front by the MOEA is excellent in this problem instance too.

However, in instance H-10-0.6-0.6 the actual Pareto front was not obtained exactly from any execution. In addition to that, all eight non-dominated sets related to different combinations of parameter values of the MOEA were under-populated, i.e. none of them consisted of 14 elements as the true Pareto front because some elements were repetitions of others. This effect can be largely attributed to the random key encoding scheme which does not exclude the possibility that two or more seemingly different chromosomes decode to the same schedule. For example, both $[0.7,0.2,0.1]$ and $[0.6,0.35,0.05]$ decode to $[1,2,3]$. The true Pareto front and the fronts returned by the MOEA for each one of the eight parameter sets are displayed in Figures 7 and 8. Note that the actual Pareto front can be obtained if the non-dominated sets related to experimental designs 2, 4 and 6 are combined. Since no locally Pareto optimal set is clearly superior to the others, the metrics described in Section 4 can assist in establishing a ranking of the eight non-dominated sets of solutions. The performance of the eight fronts regarding the metrics of generational distance (*GD*), hypervolume (*HV*), non-uniformity of solutions (*U*) and maximum spread (*MS*) are presented in Table 5. We reiterate that for the metrics of hypervolume and maximum spread high values are preferable, whereas the opposite holds for the non-uniformity of solutions and the generational distance. The information contained in Table 5 is somewhat hard to interpret and in order to facilitate a clearer presentation of the results we transform the elements of each row to the interval $[0,1]$ and construct the *spider graphs* displayed in Figure 9. Note that design 3 corresponds to the minimum values of all four performance metrics and is therefore depicted as a single point in the left pane of Figure 9. By observing Figure 9 one can sees that the local Pareto front of the third design is the best regarding the measures of *U* and *GD* but in the same time exhibits the worst performance in terms of MS and *HV*. On the other hand, the non-dominated set linked to the seventh design achieves the highest values of *HV* and *MS* but is also the worst when it comes to minimizing *GD*. All other locally Pareto optimal sets correspond to different trade-offs between the four metrics under consideration.

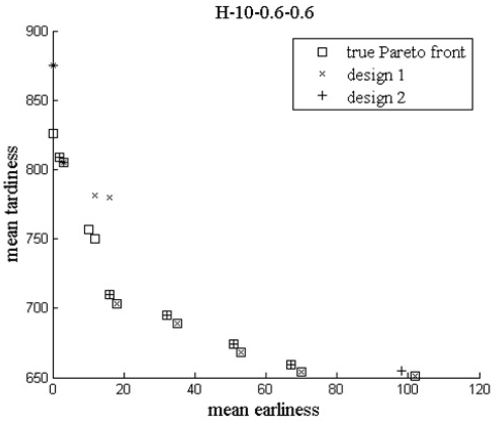


Fig. 7. True Pareto front, archives in last iteration of designs 1 and 2 – instance *H-10-0.6-0.6*

	design 1	design 2	design 3	design 4	design 5	design 6	design 7	design 8
GD	19.85	17.44	7.85	26.18	29.76	16.33	39.61	17.76
HV	18059	16948	12248	19288	20568	18323	24300	17690
U	0.67	0.75	0.45	0.49	0.59	0.54	0.65	0.56
MS	174.04	170.3	139.36	185.93	193.17	174.04	215.37	172.88

Table 5. Non-dominated sets performance metrics, instance *H-10-0.6-0.6*

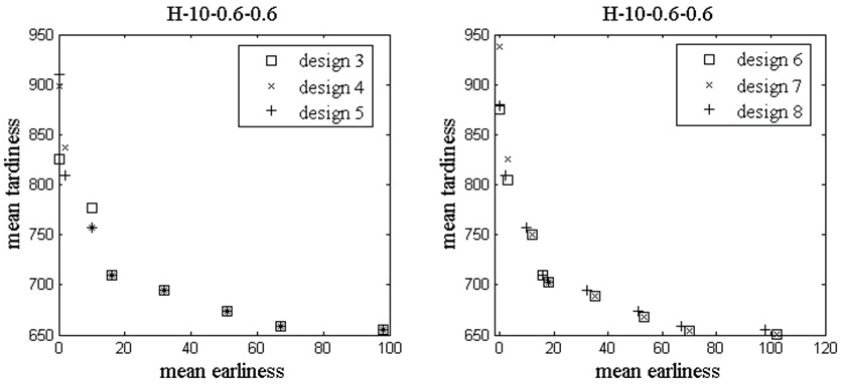


Fig. 8. Archives in last iteration of designs 3 to 8 – instance *H-10-0.6-0.6*

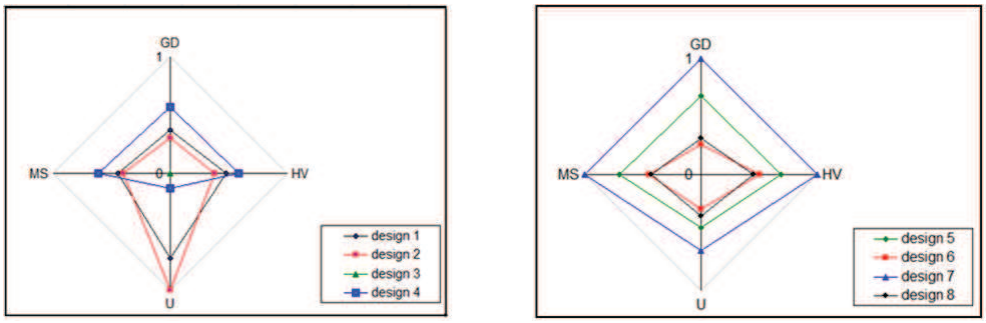


Fig. 9. Normalized performance metrics of non-dominated sets (designs 1-8) – instance *H-10-0.6-0.6*

6. Conclusion

The application of a MOEA to the multi-objective single-machine scheduling problem was investigated experimentally. The objective functions of interest were the total tardiness and the total earliness. For the purposes of this investigation the true Pareto fronts of 12 random problem instances were computed using exhaustive search. All instances consisted of ten jobs but the sizes of the Pareto fronts varied from one instance to another, indicating that the structure of the Pareto fronts does not depend only on the dimensionality of the problem. In order to tune the parameters of the MOEA to each problem instance we conducted 12 fractional factorial experiments. The use of methods from the field of Design of Experiments to select parameters for the evolutionary algorithm is appealing because they offer the ability to select an efficient parameter set with limited computational cost. The MOEA returned the actual Pareto front in ten out of twelve problem instances and computed a very good approximation of the true Pareto front in one instance. However, it exhibited a rather mediocre performance in a specific problem instance. An important advantage of the evolutionary approach to the underlying problem is that in all cases, the MOEA conducted only 1% of the number of evaluations performed by the exhaustive search algorithm. A possible direction of future research would be to apply evolution to problems with high dimensionality where the true Pareto front cannot be obtained exactly and compare the results with those from other meta-heuristic algorithms.

7. References

Bagchi, T. P. (1999). *Multiobjective scheduling by genetic algorithms*, Kluwer Academic Publishers, ISBN 0-7923-8561-6, Boston

Behnamian, J.; Fatemi Ghomi, S. M. T. & Zandieh, M. (2009). A multi-phase covering Pareto-optimal front method to multi-objective scheduling in a realistic hybrid flowshop using a hybrid metaheuristic. *Expert Systems with Applications*, Vol. 36, No. 8, October 2009, pp 11057-11069, ISSN 0957-4174

Box, G. E. P.; Hunter, W. G. & Hunter, J. S. (2005). *Statistics for Experimenters: Design, Innovation and Discovery*. Wiley-Interscience, ISBN 0-471-71813-0, New Jersey

Corne, D. W.; Jerram, N. R.; Knowles, J. D. & Oates, M.J. (2001). PESA-II: region-based selection in evolutionary multiobjective optimization, *Proceedings of the Genetic and*

Evolutionary Computation Conference (GECCO-2001), pp 283-290, ISBN 1-59593-010-8, Seattle, Washington, USA, July 2006, ACM

Corne, D. W.; Knowles, J. D. & Oates, M. J. (2000). The Pareto envelope-based selection algorithm for multiobjective optimization. *Proceedings of the Parallel Problem Solving from Nature VI Conference*, pp 839-848, ISBN 3-540-41056-2, Paris, France, September 200, Springer-Verlag, Berlin

Deb, K.; Pratap, A.; Agarwal, S. & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, Vol. 6, No. 2, April 2002, pp 182-197, ISSN 1089-778X

Everson, R. M.; Fieldsend, J. E. & Singh, S. (2002). Full elite sets for multi-objective optimization. *Proceedings of the Fifth International Conference on Adaptive Computing Design and Manufacture (ACDM 2002)*, pp 343-354, Exeter, Devon, UK, April 2002, Springer-Verlag, Berlin

Choobineh, F. F.; Mohebbi, E. & Khoo, H. (2006). A multi-objective tabu search for a single-machine scheduling problem with sequence-dependent setup times . *European Journal of Operational Research*, Vol. 175, No. 1, November 2006, pp 318-337, ISSN 0377-2217

Gupta, A. K. & Sivakumar, A. I. (2005). Multi-objective scheduling of two-job families on a single machine. *Omega*, Vol. 33, No. 5, October 2005, pp 399-405, ISSN 0305-0483

Loukil, T.; Teghem, J. & Tuyttens, D. (2005). Solving multi-objective production scheduling problems using metaheuristics. *European Journal of Operational Research*, Vol. 161, No. 1, February 2005, pp 42-61, ISSN 0377-2217

Lu, H. & Yen, G. G. (2002). Dynamic population size in multiobjective evolutionary algorithms. *Proceedings of the 2002 Congress on Evolutionary Computation*, pp 1648-1653, ISBN 0-7803-7282-4, Honolulu, Hawaii, May 2002, IEEE Computer Society, Washington DC

Moslehi, G. & Mahnam, M. (2010). A Pareto approach to multi-objective flexible job-shop scheduling problem using particle swarm optimization and local search. *International Journal of Production Economics*, In Press, Available online 10 August 2010, ISSN: 0925-5273

Pulido, G. T.; & Coello Coello, C. A. (2003). The micro genetic algorithm 2: towards online adaptation in evolutionary multiobjective optimization, *Proceedings of the Second International Conference on Evolutionary Multi-Criterion Optimization (EMO2003)*, pp 252-266, ISBN 3-540-01869-7, Faro, Portugal, April 2003, Springer-Verlag, Berlin

Tan, K. C.; Lee, T. H. & Khor, E. F. (1999). Evolutionary algorithms with goal and priority information for multi-objective optimization, *Proceedings of the 1999 IEEE International Congress on Evolutionary Computation*, Vol. 3, pp 106-113, ISBN 0-7803-5 536-9, Washington DC, USA, July 1999, IEEE Computer Society, Washington DC

Tan, K. C.; Lee, T. H. & Khor, E. F. (2005). *Multiobjective Evolutionary Algorithms and Applications*, Springer-Verlag, ISBN 1-85233-836-9, London

Valente, J. M. S. & Goncalves, J. F. (2009). A genetic algorithm approach for the single machine scheduling problem with linear earliness and quadratic

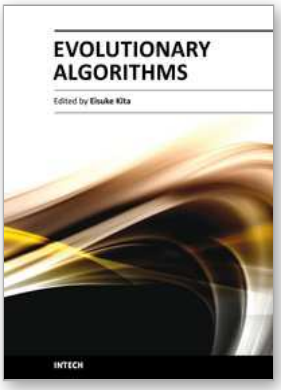
tardiness penalties. *Computers and Operations Research,* Vol. 36, No. 10, October 2009, pp 2707-2715, ISSN 0305-0548

Yagmahan, B. & Mutlu Yenisey, M. (2010). A multi-objective ant colony system algorithm for flow shop scheduling problem. *Expert Systems with Applications*, Vol. 37, No. 2, March 2010, pp 1361-1368, ISSN 0957-4174

Zitzler, E.; Laumanns, M. & Thiele, L. (2001). SPEA2: improving the strength Pareto evolutionary algorithm, Technical Report 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich

Zitzler, E.; Laumanns, M. & Thiele, L. (2002). SPEA2: improving the strength Pareto evolutionary algorithm for multiobjective optimization, In: *Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems (EUROGEN 2001),* K. Giannakoglou, D. Tsahalis, J. Periaux, K. Papailiou and T. Fogarty *(*Eds.*)*, pp 95-100

Zitzler, E. & Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation*, Vol. 3, No. 4, pp 257-271, November 1999, ISSN 1089-778X



Evolutionary Algorithms

Edited by Prof. Eisuke Kita

ISBN 978-953-307-171-8
Hard cover, 584 pages
Publisher InTech
Published online 26, April, 2011
Published in print edition April, 2011

Evolutionary algorithms are successively applied to wide optimization problems in the engineering, marketing, operations research, and social science, such as include scheduling, genetics, material selection, structural design and so on. Apart from mathematical optimization problems, evolutionary algorithms have also been used as an experimental framework within biological evolution and natural selection in the field of artificial life.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

A. S. Xanthopoulos, D. E. Koulouriotis and V. D. Tourassis (2011). Multi-Objective Scheduling on a Single Machine with Evolutionary Algorithm, Evolutionary Algorithms, Prof. Eisuke Kita (Ed.), ISBN: 978-953-307-171-8, InTech, Available from: http://www.intechopen.com/books/evolutionary-algorithms/multi-objective-scheduling-on-a-single-machine-with-evolutionary-algorithm

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821